INFORMATION AND CONTROL 15, 529–544 (1969)

Probabilistic and Weighted Grammars

ARTO SALOMAA

Mathematics Department, University of Turku, Finland

Devices for the generation of languages, corresponding to the probabilistic recognition devices or probabilistic automata, are introduced and the resulting families of languages are investigated. Comparisons are made with some other recently introduced grammars, where restrictions are imposed not only on the form of the rewriting rules but also on the use of them. A uniform representation for such grammars is provided by the notion of a grammar with a prescribed control language for the derivations.

1. INTRODUCTION

The customary Chomsky hierarchy of formal languages is obtained by imposing restrictions on the form of the rewriting rules (productions). Recently there has been an increasing amount of research done on changing the manner in which a grammar is allowed to generate words. In addition to restrictions on the form of the productions, one has imposed restrictions on the use of them. For instance, an application of some production determines which productions are applicable on the next step (this is called a programmed grammar), or some productions can never be applied if some others are applicable (an ordered grammar), or one has to apply only certain previously specified strings of productions (a matrix grammar) or, more generally, the string of productions corresponding to a derivation must belong to a set of strings previously specified (a grammar with a control set). For these notions, the reader is referred to Rosenkrantz [*4*], Friš [*2*], Âbrahám [*1*], and Ginsburg and Spanier [*3*]. Another two approaches along this line of research are time-variant and probabilistic grammars. We shall investigate the former more closely in a forthcoming paper.

In a *probabilistic grammar* there is given together with each production f a stochastic vector whose ith component indicates the probability that the i-th production is applied after f. In addition, there is given an initial probability distribution over the set of productions. In this fashion, each derivation is assigned a probability. The language generated by a probabilistic grammar consists of all words P generated by the grammar such that the probability assigned to the derivation(s) of P is greater than some previously chosen cut-point. Hereby, two interpretations will be considered. In the *maximal interpretation* it is required that there is at least one derivation of P with probability greater than the cut-point. In the *sum interpretation* it is required that the sum of the probabilities assigned to the distinct derivations of P is greater than the cut-point. The latter interpretation is customary in connection with probabilistic automata.

A probabilistic grammar is a special case of a *weighted grammar*. In the latter, vectors with arbitrary nonnegative components will be considered instead of stochastic vectors. This situation can be viewed as having a reward or punishment associated with the application of each production. The two interpretations described above will be taken into account also in connection with weighted grammars. One may impose some restrictions on probabilistic as well as on weighted grammars to guarantee the effectiveness of the procedures. A natural restriction is to assume that the probabilities and the weights are rational.

Some basic results concerning the families of languages generated by probabilistic and weighted grammars (under both maximal and sum interpretation) are established in Section 2. The corresponding families, with the additional assumption that the basic grammars are of type 3, will be studied in detail in Section 3 where also interrelations with the theory of ordinary probabilistic automata are developed. In Section 4 it is shown how certain kinds of programmed and time-variant grammars can be considered as probabilistic context-free grammars. Also all recursively enumerable sets are generated by certain modified probabilistic grammars with context-free core productions.

2. DEFINITIONS AND BASIC RESULTS

Let $G = (I_N, I_T, X_0, F)$ be a phrase structure grammar, where I_N is the set of nonterminals, I_T the set of terminals, X_0 the initial symbol and F the set of productions. Derivations according to G, the language $L(G)$ generated by G, as well as type i $(i = 0, 1, 2, 3)$ grammars in the Chomsky hierarchy obtained by imposing restrictions on F, are defined in the usual fashion, cf. Salomaa [6], pp. 164–169). Let

$$\{f_1, \cdots, f_k\} \tag{1}$$

be a set of distinct labels for the productions in F. Let

$$D: X_0 = P_0 \Rightarrow_{f_{j(1)}} P_1 \Rightarrow_{f_{j(2)}} P_2 \Rightarrow_{f_{j(3)}} \cdots \Rightarrow_{f_{j(r)}} P_r \qquad (2)$$

be a derivation according to G, where in the transition from P_i to P_{i+1} $(0 \leqq i < r)$ the production labeled by $f_{j(i+1)}$ with $1 \leqq j(i+1) \leqq k$ is applied. Then the word

$$f_{j(1)} f_{j(2)} \cdots f_{j(r)}$$

over the alphabet (1) is termed a *control word* of the derivation (2). If $r = 0$ then the control word is defined to be the empty word λ. A derivation from X_0 determines a unique control word, provided the productions in F are distinct. However, the existence of two identical productions in F is not excluded in the following discussions.

Let C be a language over the alphabet (1). (We use the term language in the most general sense to mean any set of words.) Then the language $L_C(G)$ is defined to be the subset of $L(G)$ consisting of words which possess at least one derivation whose control word is in C. $L_C(G)$ is called the language generated by G with *control language C*.

For instance, consider the grammar

$$G = (\{X, Y, Z\}, \{x, y, z\}, X, F),$$

where F consists of the productions

$$\begin{aligned} f_1 &: X \to XYZ \\ f_2 &: X \to xX \\ f_3 &: Y \to yY \\ f_4 &: Z \to zZ \\ f_5 &: X \to x \\ f_6 &: Y \to y \\ f_7 &: Z \to z. \end{aligned}$$

Assume that C consists of all words of the form

$$f_1 (f_2 f_3 f_4)^i f_5 f_6 f_7, \qquad i = 0, 1, 2, \cdots.$$

Then $L_C(G)$ is the language

$$\{x^i y^i z^i \mid i \geqq 1\}. \qquad (3)$$

The language (3) is also generated by the grammar

$$G = (\{X, Z\}, \{x, y, z\}, X, F),$$

where F consists of the productions

$$\begin{aligned} f_1 &: X \to XZ \\ f_2 &: X \to xXy \\ f_3 &: Z \to zZ \\ f_4 &: X \to xy \\ f_5 &: Z \to z, \end{aligned}$$

with the control language consisting of words of the form

$$f_1(f_2f_3)^if_4f_5, \qquad i = 0, 1, 2, \cdots.$$

Remark 1. Control languages provide a uniform way of describing grammars with restrictions on the use of productions, such as the ones mentioned in the introduction. This will be explained more closely in a forthcoming paper. Our notion of a control language differs from the notion of a control set by Ginsburg and Spanier [3] in that the latter authors restrict their attention to leftmost derivations only.

Assume that G is a type i grammar ($i = 0, 1, 2, 3$) whose productions are labeled by the labels in the set (1). Assume that φ is a mapping of the set (1) into the set of k-dimensional row vectors with nonnegative components, and that δ is a k-dimensional row vector with nonnegative components. Then the triple (G, δ, φ) is called a *weighted grammar* of type i. If, in addition, δ as well as the values of φ are stochastic vectors then (G, δ, φ) is a *probabilistic grammar* of type i. In a probabilistic grammar, δ is referred to as the initial distribution of the productions. Furthermore, the u-th component of the vector $\varphi(f_v)$, where $1 \leqq u, v \leqq k$, is referred to as the probability of applying the production labeled by f_u after applying the production labeled by f_v.

Consider a weighted grammar $G_w = (G, \delta, \varphi)$. A numerical value $\psi(D)$ will be assigned to each derivation (2), where $r > 0$. If $r = 1$ then $\psi(D)$ is defined to be the $j(1)$-th component of δ. Assume that $\psi(D)$ has been defined for the derivation (2), where $r \geqq 1$. Then, for the derivation

$$D_1 : X_0 = P_0 \Rightarrow_{j(1)} P_1 \Rightarrow_{f_{j(2)}} \cdots \Rightarrow_{f_{j(r)}} P_r \Rightarrow_{f_{j(r+1)}} P_{r+1},$$

$\psi(D_1)$ is defined to be $\psi(D)[\varphi(f_{j(r)})]_{j(r+1)}$, where the second factor is the $j(r+1)$-th component of the vector $\varphi(f_{j(r)})$.

Thus, for a probabilistic grammar $G_p = (G, \delta, \varphi)$, the number $\psi(D)$ can be interpreted as the probability of the derivation D.

Let η be a nonnegative number. We now define two languages $L_m(G_w, \eta)$ and $L_s(G_w, \eta)$ generated by the weighted grammar

$G_w = (G, \delta, \varphi)$ with *cut-point* η. The former is the subset of $L(G)$ consisting of all words which possess at least one derivation D such that

$$\psi(D) > \eta. \tag{4}$$

The latter is the subset of $L(G)$ consisting of all words P such that

$$\sum_D \psi(D) > \eta, \tag{5}$$

where D ranges over all distinct derivations of P. Thereby, two derivations are distinct if they possess distinct control words.

Assume that $L = L_m(G_w, \eta)$, for some weighted grammar $G_u = (G, \delta, \varphi)$, where G is a type i ($i = 0, 1, 2, 3$) grammar, and some cut-point η. Then we say that L is generated by a weighted grammar of type i under *maximal interpretation* or, shortly, L is w.i.m. Similarly, if $L = L_s(G_w, \eta)$, we say that L is generated by a weighted grammar of type i under *sum interpretation* or, shortly, L is w.i.s. Furthermore, if G_w is a probabilistic grammar we say that L is p.i.m. or p.i.s., respectively. Clearly, for a probabilistic grammar G_p, both $L_m(G_p, \eta)$ and $L_s(G_p, \eta)$ are empty whenever $\eta \geqq 1$.

By a *rational weighted (probabilistic) grammar* we mean a weighted (probabilistic) grammar, where the components of δ as well as the components of each value of φ are rational. If a language L is w.i.m. and, in addition, the corresponding weighted grammar and cut-point are rational, then L is said to be r.w.i.m. The abbreviations r.w.i.s., r.p.i.m. and r.p.i.s. are defined similarly.

Remark 2. As will be seen in Section 3, the sum interpretation corresponds to the interpretation customary in connection with probabilistic automata. Following the customary definition in automata theory, we have assumed a strict inequality in (4) and (5). It is a difficult problem what happens to the language families considered in this paper if in (4) and (5) the symbol $>$ is replaced by the symbol $\geqq$.

As an illustration, consider the type 2 probabilistic grammar G_p with nonterminals X and Z, terminals x, y and z, initial symbol X and the following labeled productions:

$$\begin{array}{ll}
f_1 : X \to XZ & (0, \tfrac{1}{2}, 0, \tfrac{1}{2}, 0), \\
f_2 : X \to xXy & (0, 0, 1, 0, 0), \\
f_3 : Z \to zZ & (0, \tfrac{1}{2}, 0, \tfrac{1}{2}, 0), \\
f_4 : X \to xy & (0, 0, 0, 0, 1), \\
f_5 : Z \to z & (0, 0, 0, 0, 1).
\end{array}$$

The values of φ are given together with the productions, and the initial distribution is $(1, 0, 0, 0, 0)$. Clearly, (3) is the language generated by G_p with cut-point 0 under both maximal and sum interpretation. Consequently, (3) is p.2.m. and p.2.s. and, hence, also w.2.m. and w.2.s. Since all components involved are rational, the language (3) is also r.p.2.m. r.p.2.s., r.w.2.m. and r.w.2.s.

Remark 3. Roughly speaking, in a probabilistic grammar the probabilities tend to 0 with the length of derivations. Consequently, $\eta = 0$ is the only interesting cut-point for probabilistic grammars, whereas the structure of weighted grammars is much richer. These matters will be investigated more closely later in this paper.

We shall first prove that, for each i, the family of type i languages in the Chomsky hierarchy is contained in each of the families involving i introduced above.

THEOREM 1. *For* $i = 0, 1, 2, 3$, *any language* L *of type* i *is* r.p.i.m. *and* r.p.i.s. *Consequently,* L *is* r.w.i.m., r.w.i.s., p.i.m., p.i.s., w.i.m. *and* w.i.s.

Proof. The second sentence follows from the first sentence and the definitions. To prove the first sentence, we assume that $L = L(G)$, where G is a type i grammar whose productions are labeled by the elements of the set (1). Define

$$\delta = \varphi(f_j) = (1/k, \cdots, 1/k), \qquad j = 1, \cdots, k.$$

Consider the probabilistic grammar $G_p = (G, \delta, \varphi)$. Clearly,

$$L = L_s(G_p, 0) = L_m(G_p, 0).$$

This proves the first sentence of the theorem.

Our next theorem is a lemma needed in the proofs later on. In the statement of the theorem, PQ^* denotes the language consisting of all words PQ^i, $i = 0, 1, 2, \cdots$. Thereby, P and Q are words.

THEOREM 2. *For any grammar* G, *the language* $L_C(G)$, *where the control language* C *is a finite sum of languages of the form* PQ^*, *is finite.*

Proof. Since obviously

$$L_{C+D}(G) = L_C(G) + L_D(G),$$

it suffices to consider the case where the control language C is of the form PQ^*. If Q is the empty word, then the proof is complete. Hence, we may assume that Q is not empty.

Clearly, only a finite number of derivations from the initial symbol possess the control word P and, consequently, there is only a finite number of last words in these derivations. It suffices to consider one such last word R and show that, starting from R, control words in Q^* do not lead to an infinity of terminal words.

Consider the productions labeled by the letters of Q. Let $u(v)$ be the total number of nonterminals appearing on the left (right) sides of these productions, each nonterminal being counted as many times as it occurs. If $u \leqq v$ then control words $Q^j, j \geqq 1$, do not lead to any terminal words. If $u > v$ and t is the number of nonterminals in R, then control words Q^j, where $j > t$, are not applicable. This proves the assertion.

As was seen in previous examples, Theorem 2 does not remain valid for control languages of the form $P_1Q^*P_2$. It is obviously not valid even for control languages of the form Q^*P.

THEOREM 3. *Let G_p be a probabilistic grammar of type i, $0 \leqq i \leqq 3$, and $\eta > 0$. Then the language $L_m(G_p, \eta)$ is finite.*

Proof. Assume that $G_p = (G, \delta, \varphi)$. Then

$$L_m(G_p, \eta) = L_C(G), \tag{6}$$

where the control language C is a finite sum of languages of the form PQ^*. This is seen as follows. Because $\eta > 0$, the derivation of any word belonging to the left side of (6) contains at most u transitions with probability <1, where the bound u depends on η and on the greatest probability <1 occurring in G_p. On the other hand, if a sequence of transitions with probability 1 does not constitute a loop then this sequence cannot contain more productions than the total number of productions. Clearly, if a loop (with probability 1) is entered it is impossible in the derivation to leave this loop. Consequently, if there are k distinctly labeled productions in G, then the control words of the derivations of the words in the language on the left side of (6) are of the form $PQ^j, j = 0, 1, 2, \cdots$, where the lengths of P and Q possess a finite upper bound. (In fact, the length of P does not exceed $ku + k + u$, and the length of Q does not exceed k.) Hence, (6) holds true with C of the form mentioned and Theorem 3 follows, by Theorem 2.

THEOREM 4. *For* $i = 0, 1, 2, 3$, *the family of* p.i.m. *languages is included in the family of* p.i.s. *languages. Similarly, the family of* r.p.i.m. *languages is included in the family of* r.p.i.s. *languages.*

Proof. For any probabilistic grammar G_p,

$$L_m(G_p, 0) = L_s(G_p, 0).$$

By Theorem 1, all finite languages are r.p.3.s. Theorem 4 now follows, by Theorem 3.

It is an open problem whether or not the family of w.i.m. languages is included in the family of w.i.s. languages. The same problem can be stated also for the corresponding rational families.

We shall show next how one of the decidability results concerning ordinary grammars can be extended to weighted grammars. For simplicity, we restrict ourselves to the rational case. By length-increasing productions we mean productions of the form $P \to Q$, where the length of P is less than or equal to the length of Q.

THEOREM 5. *Let* G_w *be a rational weighted grammar whose productions are length-increasing and* η *a rational number. Then there is an algorithm of deciding whether or not a given word* P *belongs to the language* $L_m(G_w, \eta)$, *and an algorithm of deciding whether or not* $P \in L_s(G_w, \eta)$.

Proof. An algorithm can be obtained by modifying the well-known algorithm for length-increasing grammars. (Cf. Salomaa [6], pp. 171–172.) In fact, the only thing different is that the occurrence of loops

$$X_0 \Rightarrow P_1 \Rightarrow \cdots \Rightarrow P_u \Rightarrow \cdots \Rightarrow P_u \Rightarrow \cdots \Rightarrow P$$

cannot be ignored. The existence of such a loop with weight >1 guarantees that P belongs to both languages under consideration. Loops with weight $\leqq 1$ can be ignored in case of maximal interpretation. Under sum interpretation, the existence of a loop with weight 1 guarantees that P belongs to the language, and the effect of a loop with weight <1 can be determined through summation.

3. PROBABILISTIC AND WEIGHTED GRAMMARS OF TYPE 3

By a *stochastic language* we mean a language acceptable by a finite probabilistic automaton with some cut-point. (The automaton is defined in the customary fashion. For instance, cf. Salomaa [6], pp. 73–77.) The automaton considered may possess an initial state or an initial distribution of states. This does not affect the family of stochastic languages.)

A language is *rational stochastic* if it is acceptable by a finite probabilistic automaton, where all of the probabilities involved are rational, with some rational cut-point. Interrelations between stochastic languages and languages generated by weighted and probabilistic grammars of type 3 will be studied in this section.

THEOREM 6. *Every stochastic language is w.3.s.*

Proof. Let L be accepted with cut-point η by the finite probabilistic automaton $A = (S, I, s_0, S_1, M)$, where S is the state set, I the input alphabet, s_0 the initial state, S_1 the final state set and M the set of transition matrices. Consider the type 3 grammar $G = (S, I, s_0, F)$, where F consists of all of the following productions:

$$s_u \to xs_v, \qquad s_u \in S, \qquad s_v \in S, \qquad x \in I; \tag{7}$$

$$s_u \to \lambda, \qquad s_u \in S_1. \tag{8}$$

Assume that k is the number of these productions. The productions are labeled by the elements of the set (1). The grammar G is extended to a weighted grammar $G_w = (G, \delta, \varphi)$ as follows. The vector δ will consist of 0's and 1's, with 1's in the positions indicating the productions with s_0 on the left side. The vector associated by φ to productions of the form (8) consists of 0's only. The vector b associated to a production of the form (7) is defined as follows. A component of b corresponding to a production with s_v on the left side equals the probability of A entering s_v, after being in s_u and receiving the input x. Other components of b equal 0. Then the transition probabilities of A are preserved in the grammar G_w and

$$L = L_s(G_w, \eta),$$

which completes the proof.

Remark 4. Let us call a language L *stochastic under maximal interpretation* if there is a finite probabilistic automaton A and a cut-point η such that L consists of words which move A from the initial state to a final state through at least one path whose probability is greater than η. Then every language stochastic under maximal interpretation is w.3.m. This is established exactly as Theorem 6.

THEOREM 7. *Let G_w be a type* 3 *weighted grammar which does not contain productions of the form $X \to Y$, where X and Y are nonterminals. Then, for any η, the language $L_s(G_w, \eta)$ is stochastic.*

Proof. Without loss of generality we may assume that the productions of G_w are of the two forms

$$X \rightarrow xY \tag{9}$$

and

$$X \rightarrow \lambda, \tag{10}$$

where X and Y are nonterminals and x is a letter of the terminal alphabet. This is seen as follows. A production of the form

$$X \rightarrow x_1x_2 \cdots x_r, \qquad r \geqq 1, \tag{11}$$

where x's are letters of the terminal alphabet, is replaced by the sequence of productions

$$X \rightarrow x_1X, \qquad X \rightarrow x_2X, \cdots, \qquad X \rightarrow x_rX, \qquad X \rightarrow \lambda. \tag{12}$$

The weights are adjusted in such a way that in the sequence (12) the transition to the next production is given weight 1 and transitions elsewhere are given weight 0. Furthermore, the transition to the first production of (12) is given the weight originally associated with the transition to the production (11). A production of the form

$$X \rightarrow x_1x_2 \cdots x_rY, \qquad r \geqq 2, \tag{13}$$

is replaced by the sequence of productions

$$X \rightarrow x_1X, \qquad X \rightarrow x_2X, \qquad \cdots, \qquad X \rightarrow x_{r-1}X, \qquad X \rightarrow x_rY, \tag{14}$$

where the weights are adjusted as above, the vector associated with the last production of (14) corresponding to the vector originally associated with (13). (Note that in a weighted grammar of type 3 the vector associated with a production $X \rightarrow P$, where P is a word over the terminal alphabet, bears no influence on the language generated.) It is clear that these changes do not affect the language $L_s(G_w, \eta)$.

Assuming that the productions of $G_w = (G, \delta, \varphi)$ are of the two forms (9) and (10), we now construct a "generalized probabilistic automaton" A, as follows. The states of A are the labels of the productions of G, the initial distribution being δ. The final state set consists of the labels of the productions (10). Consider a production of the form (9) whose label is f_u. Then the "probability" of A entering the state f_v, after being in f_u and receiving the input x, equals the v-th component of the vector associated with f_u or 0, depending on whether Y or some other nonterminal appears on the left side of the production labeled by f_v. The probability of all other transitions equals 0. (Thus, all states obtained from productions (10) are "sinks".) Then

$$L_s(G_w, \eta) = L(A, \eta),$$

where the right side denotes the language accepted by A with cut-point η. (This language is defined for A by exactly the same matrix product as for ordinary probabilistic automata.) Although A is not a probabilistic automaton, it follows by a result of Turakainen [8] that $L(A, \eta)$ is stochastic. This completes the proof.

Remark 5. It is seen from Theorems 6 and 7 that the family of stochastic languages equals the family of languages generated under sum interpretation by such type 3 weighted grammars which do not contain productions of the form $X \rightarrow Y$. It seems very likely that this restriction on the form of the productions can be removed. To do this, it suffices to prove that a language obtained from a stochastic language by deleting all occurrences of one letter is stochastic. This again is a special case of the conjecture that the family of stochastic languages is closed under homomorphism.

It is a consequence of Theorem 6 that the family of w.*i*.s. languages is nondenumerable and, therefore, contains languages which are not of type 0. This reflects the fact that no computability assumptions are made in the definitions about the real numbers involved. The following theorem is established exactly as Theorems 6 and 7.

THEOREM 8. *Every rational stochastic language is* r.w.3.s. *Let G_w be a type* 3 *rational weighted grammar which does not contain productions of the form $X \rightarrow Y$, where X and Y are nonterminals. Then, for any rational η, the language $L_s(G_w, \eta)$ is rational stochastic.*

THEOREM 9. *For a type* 3 *probabilistic grammar G_p and $\eta > 0$, the language $L_s(G_p, \eta)$ is finite. The family of* p.3.s. *languages, as well as the family of* p.3.m. *languages, equals the family of type* 3 *languages.*

Proof. The grammar G_p is first replaced by a grammar whose rules are of the forms (9), (10) and $X \rightarrow Y$, where X and Y are nonterminals. The new grammar is then rewritten as an automaton, exactly as we did in the proof of Theorem 7. Productions of the form $X \rightarrow Y$ correspond to transitions caused by the empty word. The first sentence of the theorem now follows because every loop with an exit to a final state possesses a probability less than 1. The second sentence follows from Theorems 1 and 3 and the fact that the languages $L_s(G_p, 0)$ and $L_m(G_p, 0)$ are of type 3. This is true because these languages are acceptable by finite nondeterministic automata.

Clearly, also the families of r.p.3.s. and r.p.3.m. languages equal the family of type 3 languages. However, as was seen in Section 2, the family of r.p.2.s. languages, as well as the family of r.p.2.m. languages, properly includes the family of type 2 languages.

Having discussed the families of p.3.m., p.3.s. and w.3.s. languages, we now turn to the discussion of the remaining family of w.3.m. languages.

THEOREM 10. *The family of* r.w.3.m. *languages properly includes the family of type* 3 *languages.*

Proof. The inclusion follows, by Theorem 1. Consider the rational weighted grammar G_w of type 3 with the productions

$$\begin{array}{ll} f_1 : X \to xX & (2, 1, 0) \\ f_2 : X \to yX & (0, \frac{1}{2}, 1) \\ f_3 : X \to \lambda & (0, 0, 0) \end{array}$$

and with $\delta = (1, 0, 0)$. Clearly,

$$L_m(G_w, 1) = \{x^u y^v \mid u > v \geqq 1\};$$

therefore, the inclusion is proper.

THEOREM 11. *There is an algorithm of deciding whether or not the language $L_m(G_w, \eta)$ is empty, where G_w is a rational weighted grammar of type* 3 *and η a rational number.*

Proof. We first determine all of the finitely many parts of the derivations according to G_w whose control word begins and ends with the same letter but does not have proper subwords with this property. Suppose there exists such a loop with weight >1 which, furthermore, is a part of a terminating derivation with weight >0. Then the language under consideration is not empty. Suppose no such loop exists with the described properties. Then the emptiness can be decided by checking through all of the (finitely many) derivations without loops because in this case loops do not increase the total weight of a derivation.

One can prove that there is also an algorithm of deciding whether or not the language $L_m(G_w, \eta)$ is infinite. The same problems are undecidable for the languages $L_s(G_w, \eta)$, with G_w and η as above, because the existence of a decision method would imply the decidability of the emptiness and infinity problems for rational stochastic languages. Consequently, not every r.w.3.s. language is r.w.3.m. A specific example is given in our next theorem.

THEOREM 12. *The language*

$$\{a^u b^u \mid u \geqq 1\} \tag{15}$$

is r.w.3.s. *but not* r.w.3.m.

Proof. The first assertion follows, by Theorem 8 and the results of Turakainen [7]. To prove the second assertion we assume on the contrary that (15) equals $L_m(G_w, \eta)$, for some rational weighted grammar G_w of type 3 and rational cut-point η. Without loss of generality, we may again assume that the productions of G_w are of the forms (9), (10) and $X \to Y$. Consider a word $a^v b^v$, where v exceeds the number of productions of G_w. There is a derivation of this word with weight greater than η. Furthermore, in this derivation there is a loop which begins and ends with the same production and possesses $r \geqq 1$ occurrences of productions of the form $X \to aY$. If the weight associated with this loop is >1 then the word $a^{v+r}b^v$ belongs to the language $L_m(G_w, \eta)$. If the weight is $\leqq 1$ then the word $a^{v-r}b^v$ belongs to the language. Thus, in both cases a contradiction arises and, hence, Theorem 12 follows.

We have not been able to obtain a more detailed characterization of w.3.m. languages.

4. INTERRELATIONS WITH PROGRAMMED AND TIME-VARIANT GRAMMARS

By definition, the family of w.i.s. languages is included in the family of w.j.s. languages, for $i > j$. On the other hand, by Theorem 6, the family of w.3.s. languages contains all stochastic languages. Since it is very difficult to give examples of languages which are not stochastic, it is also difficult to solve the problem of whether the families of w.i.s. languages, $i = 0, 1, 2, 3$, constitute a proper hierarchy.

The reader is referred to Rosenkrantz (1969) for a detailed definition of *programmed grammars*. In a programmed grammar, the productions are labeled and together with each production there are given two sets of labels: the *success field* and the *failure field*. After the application of some production f, only productions with labels in the success field of f are applicable on the next step of the derivation. If f is not applicable, the next production applied must have its label in the failure field of f. A remarkable result established by Rosenkrantz [*4*] is that all recursively enumerable (i.e., type 0) languages are generated by programmed grammars with context-free (i.e., type 2) core productions.

From the point of view of weighted (and time-variant) grammars, programmed grammars with context-free core productions and with *empty failure fields* are of special interest. We do not know of any characterization of the family of languages generated by programmed grammars of the described kind. However, it is easy to show that this family contains all context-sensitive (i.e., type 1) languages, provided the result by Ábrahám (1965) is correct.

THEOREM 13. *Any language generated by a programmed grammar G with context-free core productions and with empty failure fields is* r.p.2.m. *and* r.p.2.s.

Proof. The given programmed grammar G is transformed into a rational probabilistic grammar G_p of type 2, as follows. If the success field of a production f contains $r \geqq 1$ labels then the transition from f to each of these labels is given probability $1/r$, and the transition from f to all other labels is given probability 0. If the success field of f is empty then a "sink" production $X \to X$ is added to the grammar, and the transition from f to this sink production is given probability 1. (A common sink production may be used for all productions of G with empty success fields.) The initial distribution δ is defined similarly. Then both of the languages $L_s(G_p, 0)$ and $L_m(G_p, 0)$ equal the language generated by G and, hence, Theorem 13 follows.

Remark 6. Consider a type i ($i = 0, 1, 2, 3$) grammar

$$G = (I_N, I_T, X_0, F).$$

The grammar G together with an infinite sequence

$$F_1, F_2, F_3, \cdots \tag{16}$$

of subsets of F forms a *time-variant grammar* G_{tv} of *type* i. (Cf. Salomaa [*5*] for a corresponding recognition device of type 3.) The language $L(G_{tv})$ generated by G_{tv} is defined to be $L_C(G)$, where the control language C is the union of all languages of the form $F_1F_2 \cdots F_u$, $u \geqq 1$, where juxtaposition denotes catenation. We will discuss time-variant grammars more closely in a forthcoming paper. Of special interest are such type 2 time-variant grammars, where the sequence (16) is periodic.

Languages generated by these grammars are called *periodically time-variant context-free* languages or, shortly, p.t.v.c.f. [It can be shown that the same languages are obtained by assuming that the sequence (16) is almost periodic.] For instance, consider the type 2 grammar with the initial symbol X_0 and the productions

$$\begin{aligned} &f_1 : X_0 \to xX_1yX_2 \\ &f_2 : X_1 \to xX_1y \\ &f_3 : X_1 \to \lambda \\ &f_4 : Y_2 \to \lambda \\ &f_5 : X_2 \to zX_2 \\ &f_6 : X_2 \to Y_2 . \end{aligned}$$

Time-variance is specified in such a way that, for $u = 1, 2, \cdots$, the productions f_1–f_4 belong to the set F_{2u-1} and f_5–f_6 belong to the set F_{2u}. Then the language generated is (3) which, thus, is p.t.v.c.f. (Note that Y_2 is introduced to prevent the derivation of words $x^{i+1}y^{i+1}z^i$.) Although derivations according to p.t.v.c.f. grammars are easy to describe and perform, the generative power of these grammars is remarkable. It can be shown that the family of p.t.v.c.f. languages includes all languages generated by context-free matrix grammars. Hence, the family of p.t.v.c.f. languages contains all context-sensitive languages, provided the result by Ábrahám (1965) is correct. On the other hand, p.t.v.c.f. languages are a subset of the family of languages generated by programmed grammars with context-free core productions and empty failure fields. This subset is obtained by imposing on programmed grammars the further restriction that whenever two labels f_1 and f_2 are in the success field of a production then the productions labeled by f_1 and f_2 possess identical success fields.

Throughout this paper, we have assumed in considering a step

$$P_1 \Rightarrow_f P_2 \tag{17}$$

of a derivation that the production f is actually applied, i.e., $P_1 = Q_1QQ_2$, $P_2 = Q_1RQ_2$ and f is the label of the production $Q \to R$, for some Q_1 and Q_2. Another possibility is to specify a subset F_1 of productions such that the notation (17) may be used *also* in case $f \in F_1$ is not applicable, i.e., P_1 does not contain an occurrence of Q and $P_1 = P_2$. Let us assume that this possibility is included in (2) when control words are defined. Everything concerning weighted and probabilistic grammars is defined now as it was defined before using this new interpretation, the so-called *checking interpretation*, of control words. Then Theorem 13 can be strengthened to the following:

THEOREM 14. *Every recursively enumerable language is* r.p.2.m. *and* r.p.2.s., *under checking interpretation.*

Proof. By the result of Rosenkrantz (1969), any recursively enumerable language L is generated by a programmed grammar G with context-free core productions. We first rewrite G so that, for each production f, either the success field of f or the failure field of f is empty. This is done by making two copies of f, one with the success field and the other with the failure field of the original f, and putting the labels of both copies into all fields which contain the label of the original production f. We then replace all productions $X \rightarrow P$ with an empty success field by productions $X \rightarrow Y$, where Y is a new nonterminal (which, thus, does not appear on the left side of any production). The rest of the proof proceeds like the proof of Theorem 13.

It is always possible in Theorem 14 to choose the required rational probabilistic type 2 grammar $G_p = (G, \delta, \varphi)$ in such a way that the language $L(G)$ generated by the basic grammar G is of type 3.

RECEIVED: June 4 1969

REFERENCES

1. S. ÂBRAHÁM, Some questions of phrase structure grammars, *Computational Linguistics* 4 (1965), 61–70.
2. I. FRIŠ, Grammars with partial ordering of the rules, *Information and Control* **12** (1968), 415–425.
3. S. GINSBURG, AND E. H. SPANIER, Control sets on grammars, *Math. Systems Theory* **2** (1968), 159–177.
4. D. J. ROSENKRANTZ, Programmed grammars and classes of formal languages, *J. Assoc. Comput. Mach.* **16** (1969), 107–131.
5. A. SALOMAA, On finite automata with a time-variant structure, *Information and Control* **13** (1968), 85–98.
6. A. SALOMAA, "Theory of Automata," Pergamon Press, Oxford, 1969.
7. P. TURAKAINEN, On languages representable in rational probabilistic automata, *Ann. Acad. Sci. Fenn.* (1969), *Ser.* AI, 439.
8. P. TURAKAINEN, Generalized automata and stochastic languages. *Proc. Amer. Math. Soc.* **21** (1969), 303–309.